MULTIMEDIA UNIVERSITY

STUDENT ID NO

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 1, 2019/2020

ECE4246 – ADVANCED OBJECT-ORIENTED DESIGN WITH JAVA

(All Sections / Groups)

15 OCTOBER 2019
9:00 A.M – 11:00 A.M.
(2 Hours)

INSTRUCTIONS TO STUDENT

1. This examination paper consists of SIX (6) pages including cover with THREE (3) questions only.
2. Answer ALL questions. All questions carry 25 marks and the distribution of the marks for each question is given.
3. This examination is OPEN-BOOK. Allowed material: Textbook and/or bound notes only.
4. Please print all your answers in the Answer Booklet provided.

Question 1

Figure Q1a shows the code of a class, `Area`.

```
public class Area
{
	private String name;
	private String postcode;

	public Area (String nm, String pr)
	{
		setName (nm);
		setPostcode (pr);
	}

	public void setName (String nm) { name = nm;	}

	public String getName () {	return name;	}

	public void setPostcode (String pr) { postcode = pr; }

	public String getPostcode () { return postcode; }

	public String toString () { return name + "\tPostcode: " +
postcode;	}
}
```

Figure Q1a: Class `Area`

Write a Java application class `AreaSearch` with the following requirements:

(a) It should accept the names of locations and their postcode and save them into an `ArrayList` of `Area` objects. Allow the user to enter multiple locations repeatedly. After user enters the input 'n', indicating no more locations to input, display the list of locations and their postcodes. See Figure Q1b.

[14 marks]

(b) After user enters the input 'n', prompt user for an area name to search for. Search for it in the list.

[3 marks]

(c) If found, display "Search Successful" and print the name of the area along with its postcode, or else display the message "Not found in list.". See Figures Q1b and Q1c.

[5 marks]

(d) Handle all exceptions that may occur.

[3 marks]

Continued...

```
Enter Area: Alor
Enter postcode: 05600
Another? (y/n): y
Enter Area: Kulai
Enter postcode: 81000
Another? (y/n): n
Area : Alor     Postcode : 05600
Area : Kulai    Postcode : 81000
Enter area name to search:
Cyberjaya
Not found in list.
```

Figure Q1b: Sample output

```
Enter Area: Alor Setar
Enter postcode: 05700
Another? (y/n): y
Enter Area: Bagan
Enter postcode: 83000
Another? (y/n): y
Enter Area: Cyberjaya
Enter postcode: 234hi
Invalid input
Another? (y/n): y
Enter Area: Cyberjaya
Enter postcode: 63000
Another? (y/n): n
Area : Alor Setar       Postcode : 05700
Area : Bagan    Postcode : 83000
Area : Cyberjaya        Postcode : 63000
Enter area name to search: Bagan
Search successful
Area : Bagan    Postcode : 83000
```

Figure Q1c: Sample output

Continued...

Question 2

(a) Write an abstract class, **`Lifeform`**, that represents a general organism. It has an instance variable, **`name`** (type **`String`**).
- Provide a parameterized constructor, mutator and accessor methods.
- Override the method **`clone()`** from **`Object`** to return a copy of this object.
- Include an abstract method, **`speak()`**, which returns a **`String`** value.

[8 marks]

(b) Write two concrete classes, **`Human`** and **`Horse`**, which inherit from **`Lifeform`**. These classes do not create additional instance variables.
- Override the superclass's **`clone`** method.
- Implement the superclass's abstract method (*Hint*: return descriptions of how the class objects speak. For example, a **`Human`** object's **`speak()`** may return the value "Hello").

[6 marks]

(c) Write an interface, **`WorkingAnimal`**, which represents the behaviours of a class that becomes a working animal. The interface has two methods: **`ownedBy(Human)`** and **`doWork(String)`**. Both methods do not return any values.

[2 marks]

(d) Write a class, **`WorkHorse`**, which inherits from **`Horse`** and implements the interface **`WorkingAnimal`**. The class has two instance variables, **`owner`** (type **`Human`**) and **`task`** (type **`String`**).
- Provide a constructor. Instance variable **`owner`** should be initialized to **`null`** and **`task`** (which is a task the working animal is supposed to do) to an empty string.
- Implement all the abstract methods of the interface **`WorkingAnimal`**. Method **`ownedBy`** assigns a **`Human`** as the owner and **`doWork`** assigns a new task to **`task`**.

[6 marks]

(e) Draw a Unified Modeling Language (UML) class diagram, displaying only the class names (**DO NOT** list any of the members of the classes) to show the relationships among **`Lifeform`**, **`Human`**, **`Horse`**, **`WorkingAnimal`**, and **`WorkHorse`**.

[3 marks]

Note: **DO NOT write a driver class.**

Continued...

Question 3

Figure Q3a shows the UML class diagram of an interface, **`ExtraPay`**. This interface defines two methods, **`calcBonus`** and **`calcOT`**. Figure Q3b shows the class diagram of a class, **`Personnel`**, which represents the person employed by the company, in terms of first and last names.

(*Note*: **DO NOT write `ExtraPay` and `Personnel`**)

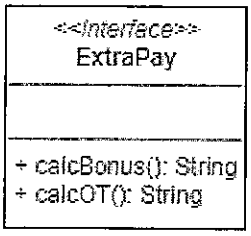

Figure Q3a

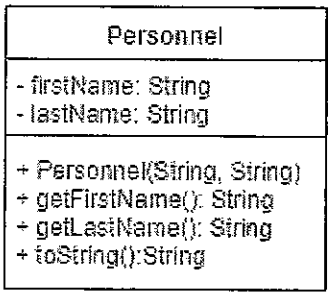

Figure Q3b

(a) Write a class, **`Employee`**, which implements **`ExtraPay`**. It has four instance variables: **`employee`** (type **`Personnel`**), **`jobtitle`** (type **`String`**), **`empid`** (type **`int`**) and **`OThours`** (type **`int`**). The variables have **`protected`** access and represent the name of the employee, the job title held, the employee's unique identity number and the number of overtime hours they have worked.

- Provide a constructor which accepts arguments to initialize all instance variables.
- Provide a **`toString`** method. This method returns a description of the employee and the employee's extra pay, by invoking the abstract methods of **`ExtraPay`**. (*Note*: For simplicity's sake, it is not necessary to provide accessors or mutators for this class)

[5 marks]

(b) Write a class, **`Permanent`** (which represents a permanent employee), which inherits from **`Employee`**. It has three variables: **`salary`** (type **`double`**), **`performancelvl`**(type **`int`**) and **`bonus`**(type **`double`**).

- Provide a constructor which accepts arguments to initialize all the class's instance variables.
- Override the superclass **`toString`** method.
- Implement the superclass abstract methods **`calcBonus`** and **`calcOT`**, both of which return **`String`** representations of the bonus and overtime (OT). Permanent employees receive an annual bonus based on their performance level (1-3). Level 1 receives no bonus, level 2 receives one month while level 3 receives 2 months. The overtime is also calculated at RM100 per hour. Figure Q3c shows the output (of **`toString`**) of a sample **`Permanent`** object.

[8 marks]

Continued...

(c) Write a class, **Contract** (which represents a contract employee), which inherits from **Employee**. It has three variables: **startdate** (type **String**), **enddate** (type **String**) and **contractpay** (type **double**).

- Provide a constructor which accepts arguments to initialize all the class's instance variables.
- Override the superclass **toString** method.
- Implement the superclass abstract methods **calcBonus** and **calcOT**, both of which return **String** representations of the bonus and overtime (OT). Contract employees receive a one-off bonus based on 25% of their contract pay. The overtime is also calculated at RM60 per hour. Figure Q3d shows the output (of **toString**) of a sample **Contract** object.

[8 marks]

(d) Identify the design pattern implemented in the design of **ExtraPay**, **Employee**, **Permanent**, and **Contract**. Briefly describe the context, in terms of the said interface / classes, and its corresponding solution.

[4 marks]

```
Permanent Employee:
Employee name: Jim Parsons
Employee ID: 12252
Job title: Physicist
Overtime Hours Worked: 25

Salary: 5230.0
Performance Level: 3

Bonus: RM10460.0
Overtime Pay: RM2500
```

Figure Q3c: Sample **toString** output of a **Permanent** object

```
Contract Employee:
Employee name: Lenard Nimoy
Employee ID: 22521
Job title: Vulcan Ambassador
Overtime Hours Worked: 30

Contract Pay: 35000.0
Start Date - End Date: 25 January 2019-31 August 2019
Bonus: RM8750.0
Overtime Pay: RM1800
```

Figure Q3d: Sample **toString** output of a **Contract** object

End of Paper